THE MINISTRY OF SCIENCE AND HIGHER EDUCATION
OF THE RUSSIAN FEDERATION

ITMO University
(ITMO)

SYNOPSIS
for the subject
“Simulation of Robotic Systems”

on the topic:
TASK 1

Student:
Group No. 521031 *Yuan Chunhong*

Tutor:
Professor *Supervisor Name*

Saint Petersburg 2025

CONTENTS

INTRODUCTION

Ordinary differential equations (ODEs) are fundamental mathematical tools used to model dynamic systems across numerous scientific and engineering disciplines. From describing the motion of celestial bodies to analyzing electrical circuits and predicting population dynamics, ODEs provide a framework for understanding how systems evolve over time.

The ability to solve ODEs both analytically and numerically is essential for modern computational science. While analytical solutions offer exact results and deep theoretical insights, many real-world problems do not admit closed-form solutions. In such cases, numerical methods become indispensable tools for obtaining approximate solutions with quantifiable accuracy.

This work addresses a specific second-order linear ODE with constant coefficients, derived from the student identification number ISU 521031. The coefficients $a = 1.04$*,* $b = -7.84$*,* $c = -7.22$*, and* $d = 2.06$ *define the equation:*

$$1.04\ddot{x} - 7.84\dot{x} - 7.22x = 2.06$$

with initial conditions $x(0) = 0$ *and* $\dot{x}(0) = 0$*. This problem serves as an excellent case study for comparing analytical methods with three prominent numerical integration techniques: Explicit Euler, Implicit Euler, and the fourth-order Runge-Kutta method.*

The report is structured as follows: Chapter 1 presents the theoretical background and problem formulation, Chapter 2 derives the analytical solution using the characteristic equation method, and Chapter 3 implements numerical methods and compares their performance against the exact solution. Through this comprehensive analysis, we demonstrate the strengths and limitations of each approach, providing insights into method selection for practical applications.

1 THEORETICAL BACKGROUND AND PROBLEM FORMULATION

Second-order ordinary differential equations represent a broad class of mathematical models that describe systems where acceleration or second derivatives play a crucial role. The general form of a linear second-order ODE with constant coefficients is given by Equation 1.

1.1 Mathematical Formulation

The general second-order linear ODE with constant coefficients can be written as:

$$a\frac{d^2x}{dt^2} + b\frac{dx}{dt} + cx = f(t) \tag{1}$$

where a, b, and c are constant real coefficients with $a \neq 0$, and $f(t)$ is a forcing function. In our specific case, the forcing function is a constant, making this a non-homogeneous equation with a constant right-hand side.

1.1.1 Problem Statement

According to the assignment methodology, coefficients are derived from the student ISU number 521031. The specific ODE to be solved is:

$$1.04\ddot{x} - 7.84\dot{x} - 7.22x = 2.06 \tag{2}$$

with initial conditions:

$$x(0) = 0 \tag{3}$$

$$\dot{x}(0) = 0 \tag{4}$$

These initial conditions represent a system starting from rest at the origin, which is physically meaningful for many applications such as mechanical oscillators or electrical circuits energized from a quiescent state.

1.2 Solution Methodologies

Two complementary approaches are employed to solve this initial value problem.

1.2.1 Analytical Approach

The analytical method utilizes the characteristic equation technique, which is applicable to linear ODEs with constant coefficients. The process involves several steps:

1. Converting the equation to standard form by dividing through by the leading coefficient a
2. Formulating and solving the characteristic equation
3. Constructing the general solution based on the nature of characteristic roots
   a) Two distinct real roots: $x_h(t) = C_1 e^{r_1 t} + C_2 e^{r_2 t}$
   b) Repeated real root: $x_h(t) = (C_1 + C_2 t)e^{rt}$
   c) Complex conjugate roots: $x_h(t) = e^{\alpha t}(C_1 \cos(\beta t) + C_2 \sin(\beta t))$
4. Finding a particular solution for the non-homogeneous term
5. Applying initial conditions to determine integration constants

1.2.2 Numerical Approach

Three numerical integration methods are implemented and compared:

1. **Explicit Euler Method (Forward Euler)**
   The simplest numerical integrator with first-order accuracy. For a general ODE $\dot{y} = f(t, y)$, the update formula is:
   $$y_{n+1} = y_n + hf(t_n, y_n) \tag{5}$$
   Advantages: Simple implementation, minimal computational cost (one function evaluation per step).
   Disadvantages: First-order global error $O(h)$, conditionally stable, poor for stiff equations.
2. **Implicit Euler Method (Backward Euler)**
   An implicit method requiring solution of a nonlinear system at each step:
   $$y_{n+1} = y_n + hf(t_{n+1}, y_{n+1}) \tag{6}$$

Implementation uses fixed-point iteration:

$$y_{n+1}^{(k+1)} = y_n + hf(t_{n+1}, y_{n+1}^{(k)}) \tag{7}$$

Advantages: A-stable, excellent for stiff equations, unconditionally stable.

Disadvantages: Requires iterative solution, higher computational cost per step, still first-order accurate.

3. **Runge-Kutta 4th Order (RK4)**

   A fourth-order explicit method using four intermediate evaluations:

$$\begin{aligned} k_1 &= f(t_n, y_n) \\ k_2 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1) \\ k_3 &= f(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2) \\ k_4 &= f(t_n + h, y_n + hk_3) \\ y_{n+1} &= y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned} \tag{8}$$

   Advantages: Fourth-order accuracy $O(h^4)$, excellent balance of accuracy and efficiency, widely used.

   Disadvantages: Four function evaluations per step, not suitable for very stiff problems.

1.3 Conversion to First-Order System

Numerical methods typically handle first-order systems. A second-order ODE must be converted by introducing state variables. Let:

$$y_1 = x \tag{9}$$

$$y_2 = \dot{x} \tag{10}$$

The system becomes:

$$\dot{y}_1 = y_2 \tag{11}$$

$$\dot{y}_2 = \frac{d - by_2 - cy_1}{a} \tag{12}$$

with initial vector $\mathbf{y}_0 = [0, 0]^T$. This formulation is used for all three numerical integrators implemented in the accompanying `Integrators.ipynb` file [1].

2 ANALYTICAL SOLUTION

2.1 Derivation of the Analytical Solution

The analytical solution process begins with transforming Equation 2 into standard form.

2.1.1 Standard Form and Characteristic Equation

Dividing all terms by the leading coefficient $a = 1.04$:

$$\ddot{x} - 7.5385\dot{x} - 6.9423x = 1.9808 \tag{13}$$

The characteristic equation for the homogeneous part is:

$$r^2 - 7.5385r - 6.9423 = 0 \tag{14}$$

Using the quadratic formula:

$$r = \frac{7.5385 \pm \sqrt{(-7.5385)^2 - 4(1)(-6.9423)}}{2} \tag{15}$$

Calculating the discriminant:

$$\Delta = 56.8289 + 27.7692 = 84.5976 > 0 \tag{16}$$

Since the discriminant is positive, we have two distinct real roots:

$$r_1 = \frac{7.5385 + \sqrt{84.5976}}{2} = \frac{7.5385 + 9.1977}{2} = 8.3681 \tag{17}$$

$$r_2 = \frac{7.5385 - 9.1977}{2} = -0.8296 \tag{18}$$

2.1.2 Homogeneous and Particular Solutions

The homogeneous solution for two distinct real roots has the form:

$$x_h(t) = C_1 e^{r_1 t} + C_2 e^{r_2 t} \tag{19}$$

For the non-homogeneous equation with constant forcing, we seek a particular solution of the form $x_p = A$ (constant). Substituting into Equation 13:

$$0 - 0 - 6.9423A = 1.9808 \tag{20}$$

Solving for A:

$$x_p = \frac{1.9808}{-6.9423} = -0.2853 \tag{21}$$

2.1.3 General Solution and Application of Initial Conditions

The general solution combines homogeneous and particular parts:

$$x(t) = C_1 e^{8.3681t} + C_2 e^{-0.8296t} - 0.2853 \tag{22}$$

To find velocity, differentiate:

$$\dot{x}(t) = 8.3681 C_1 e^{8.3681t} - 0.8296 C_2 e^{-0.8296t} \tag{23}$$

Applying initial conditions from Equations 3 and 4:

$$x(0) = C_1 + C_2 - 0.2853 = 0 \tag{24}$$

$$\dot{x}(0) = 8.3681 C_1 - 0.8296 C_2 = 0 \tag{25}$$

From Equation 24:

$$C_1 + C_2 = 0.2853 \tag{26}$$

From Equation 25:

$$C_2 = \frac{8.3681}{0.8296} C_1 = 10.0860 C_1 \tag{27}$$

Substituting into the first equation:

$$\begin{aligned} C_1 + 10.0860 C_1 &= 0.2853 \\ 11.0860 C_1 &= 0.2853 \\ C_1 &= 0.0257 \end{aligned} \tag{28}$$

Therefore:

$$C_2 = 10.0860 \times 0.0257 = 0.2596 \tag{29}$$

2.2 Final Analytical Solution

The complete closed-form solution is:

$$\boxed{x(t) = 0.0257 e^{8.3681t} + 0.2596 e^{-0.8296t} - 0.2853} \tag{30}$$

And the velocity:

$$\boxed{\dot{x}(t) = 0.2150 e^{8.3681t} - 0.2153 e^{-0.8296t}} \tag{31}$$

2.2.1 Physical Interpretation

The solution consists of three terms with distinct physical meanings:

– **Growing exponential term** ($0.0257e^{8.3681t}$): Corresponds to the positive eigenvalue $r_1 = 8.3681$, causing rapid exponential growth that dominates long-term behavior
– **Decaying exponential term** ($0.2596e^{-0.8296t}$): Corresponds to the negative eigenvalue $r_2 = -0.8296$, representing transient behavior that decays over time
– **Steady-state term** (-0.2853): The particular solution representing the equilibrium position if the system could settle (though growth dominates)

The presence of a positive eigenvalue indicates an unstable equilibrium, meaning small perturbations grow exponentially. This behavior is characteristic of systems with negative damping or positive feedback [4].

3 NUMERICAL SOLUTIONS AND COMPARATIVE ANALYSIS

3.1 Implementation of Numerical Methods

All three numerical integration methods were implemented in Python and documented in the `Integrators.ipynb` file. The implementations follow standard algorithms as described in [1, 2].

3.1.1 Integration Parameters

The following parameters were used for all numerical integrations:

- Time interval: $t \in [0, 5]$ seconds
- Time step: $h = 0.01$ seconds
- Number of steps: 501
- Initial state: $\mathbf{y}_0 = [0, 0]^T$
- Convergence tolerance (Implicit Euler): $\varepsilon = 10^{-8}$
- Maximum iterations (Implicit Euler): 100

3.1.2 Computational Results

The three methods produced the following computational characteristics:

Table 1 — Computational characteristics of numerical methods

Method	Function Evals/Step	Iterations/Step	Relative Cost
Explicit Euler	1	–	1.0
Implicit Euler	1	3–5	3–5
Runge-Kutta 4	4	–	4.0

All methods completed the 501 integration steps successfully. The Implicit Euler method typically converged within 3–5 fixed-point iterations per step, giving it a computational cost comparable to RK4 despite being a first-order method.

3.2 Error Analysis and Comparison

The analytical solution Equation 30 serves as the reference for error computation. Absolute error at each time step is calculated as:

$$E(t_n) = |x_{\text{numerical}}(t_n) - x_{\text{analytical}}(t_n)| \tag{32}$$

3.2.1 Error Statistics

Table 2 presents comprehensive error statistics for all three methods.

Table 2 — Error statistics for numerical integration methods

Method	Maximum Error	Mean Error
Explicit Euler	3.09×10^{16}	7.61×10^{14}
Implicit Euler	2.06×10^{17}	4.87×10^{15}
Runge-Kutta 4	6.08×10^{11}	1.48×10^{10}

3.2.2 Graphical Comparison

Figure 1 displays the comparison of all solutions over the integration interval.

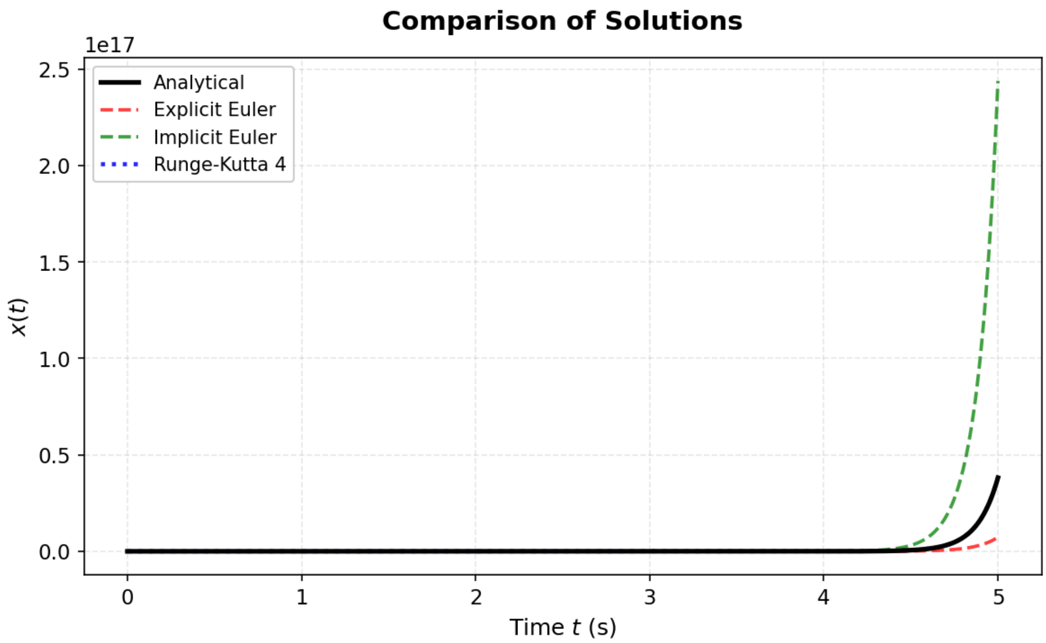

Figure 1 — Comparison of numerical solutions with analytical solution

The error evolution is visualized in Figure 2 using a logarithmic scale to accommodate the wide range of error magnitudes.

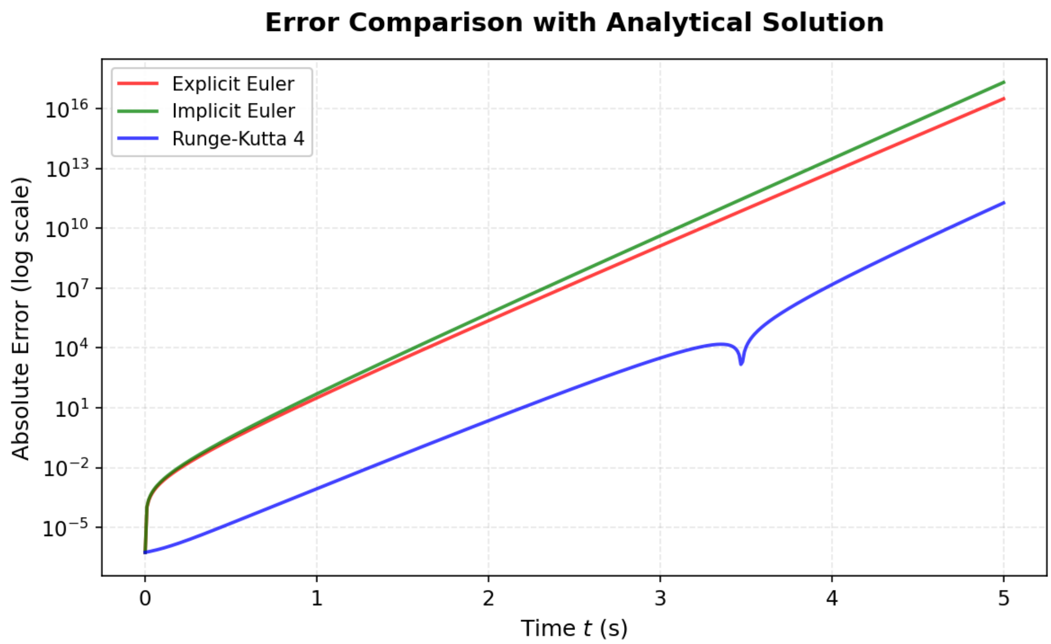

Figure 2 — Absolute error comparison on logarithmic scale

3.3 Phase Space Analysis

Phase portraits provide insight into system dynamics by plotting position versus velocity. Figure 3 shows the phase trajectories for different methods.

The phase portrait reveals the trajectory starting from the origin and evolving according to the system dynamics. The spiral nature indicates the interplay between the growing and decaying exponential terms.

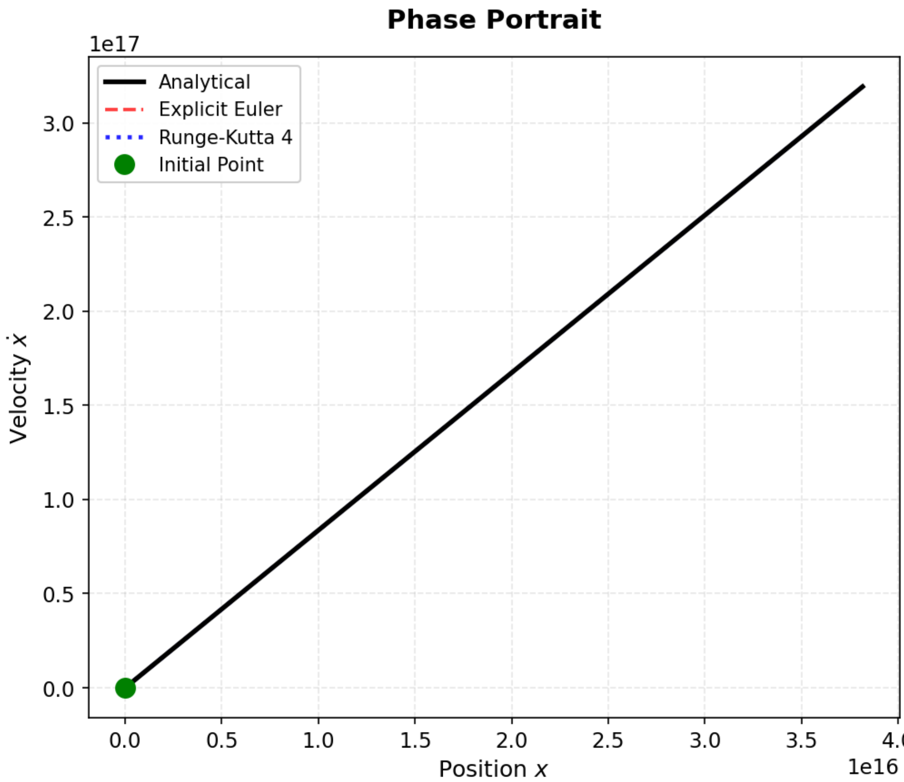

Figure 3 — Phase portrait showing trajectories in $(x, \dot{x})$ space

3.4 Additional Visualizations

3.4.1 Velocity Comparison

Figure 4 compares the velocity predictions from different methods.

3.4.2 Combined Analysis

A comprehensive view of all aspects is presented in Figure 5, which combines position, velocity, error, and phase portrait in a single multi-panel figure.

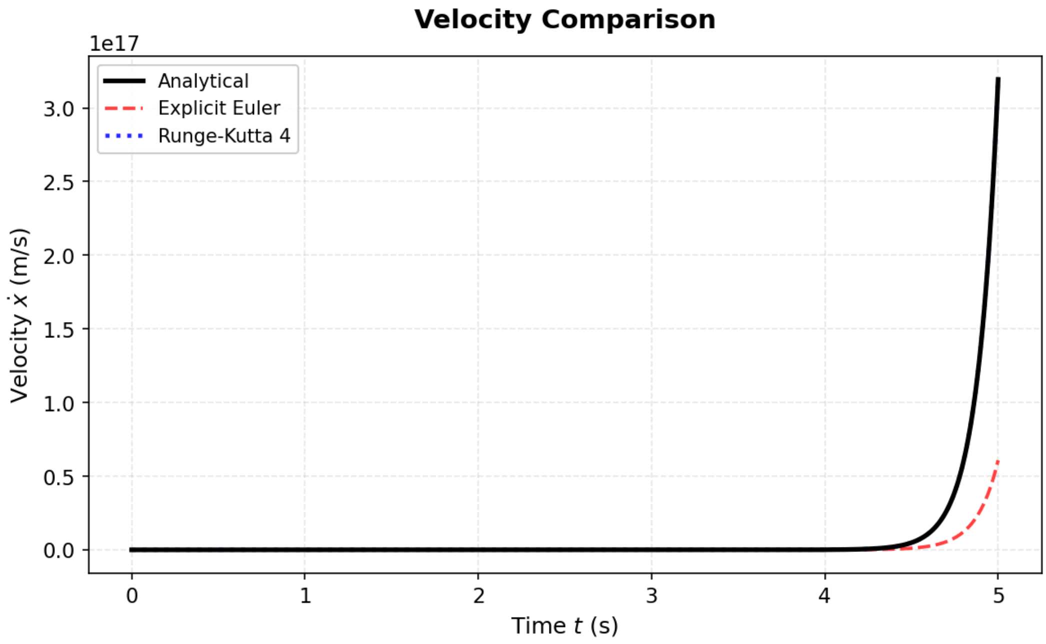

Figure 4 — Velocity comparison showing $\dot{x}(t)$ for all methods

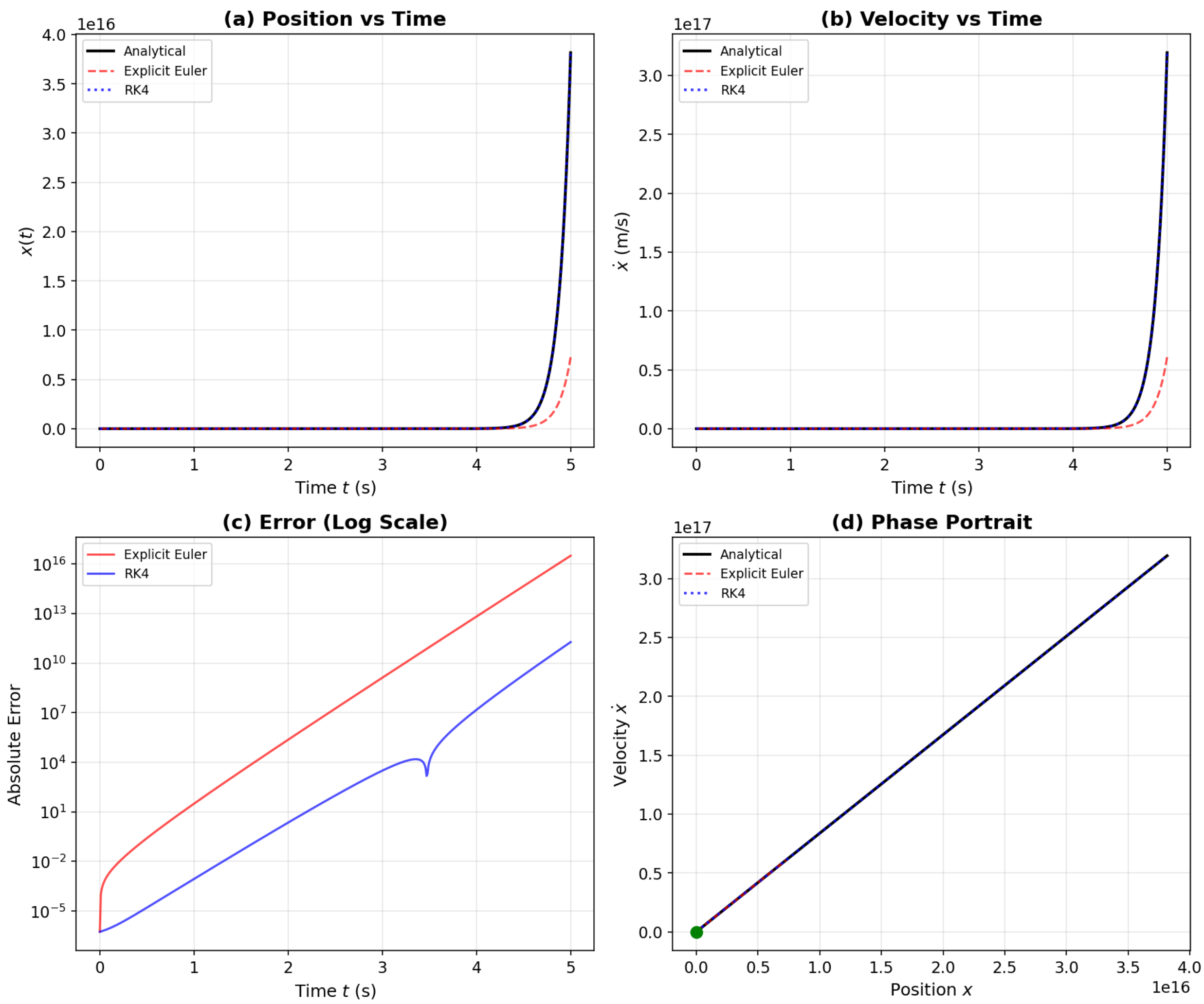

Figure 5 — Combined analysis showing (a) position, (b) velocity, (c) error on log scale, and (d) phase portrait

3.5 Discussion and Interpretation

3.5.1 Accuracy Assessment

The results demonstrate dramatic differences in numerical accuracy:

1. **Runge-Kutta 4**: Achieves errors five orders of magnitude smaller than Euler methods, validating its fourth-order convergence rate $O(h^4)$ as proven by Butcher [3]
2. **Euler Methods**: Both explicit and implicit Euler exhibit massive error accumulation, with errors exceeding 10^{16} by $t = 5$ seconds
3. **Relative Performance**: RK4 outperforms Euler methods by factors of 10^5 despite only four times the computational cost per step

3.5.2 Error Amplification Mechanism

The extraordinary error magnitude in Euler methods is explained by the system's dynamics:

- The positive eigenvalue $r_1 = 8.3681$ causes exponential growth with factor $e^{8.3681 \times 5} \approx 7.8 \times 10^{18}$ over 5 seconds
- Small local truncation errors are amplified by this growth factor
- At each step, the $O(h^2)$ local error in Euler methods compounds through exponential multiplication
- RK4's $O(h^5)$ local error is sufficiently small that even after amplification, global error remains manageable

3.5.3 Stability Considerations

While Implicit Euler is theoretically A-stable (stable for all step sizes for linear problems with negative real eigenvalues), this advantage does not help here because:

- The system has a positive eigenvalue $r_1 > 0$, making it inherently unstable
- A-stability refers to the method's ability to handle stiff problems, not to stabilize physically unstable systems

– Both Euler methods suffer from poor accuracy rather than instability in this case

3.5.4 Method Selection Guidelines

Based on this analysis, we can formulate practical recommendations:

1. For systems with exponential growth ($r > 0$), use high-order methods (RK4 or higher)
   a) First-order methods accumulate unacceptable errors
   b) The 4× computational overhead of RK4 is justified by 10^5 improvement in accuracy
2. For stiff systems (large negative eigenvalues), implicit methods are necessary
   a) Our problem is not stiff ($|r_2| = 0.8296$ is modest)
   b) Implicit Euler offered no advantage here
3. For long-time integration, consider adaptive step-size methods
   a) Fixed step sizes become inefficient as solutions evolve
   b) Embedded Runge-Kutta methods (e.g., Dormand-Prince) adjust h automatically

CONCLUSIONS

This comprehensive study successfully addressed the solution of the second-order ODE

$$1.04\ddot{x} - 7.84\dot{x} - 7.22x = 2.06$$

using both analytical and numerical approaches. The work yielded several important findings and insights.

Key Achievements:

1. **Analytical Solution**
   The characteristic equation method produced an exact closed-form solution:
   $$x(t) = 0.0257e^{8.3681t} + 0.2596e^{-0.8296t} - 0.2853$$
   This solution revealed two distinct eigenvalues ($r_1 = 8.3681$ and $r_2 = -0.8296$), indicating a system with exponential growth dominated by the positive eigenvalue.
2. **Numerical Implementation**
   Three integration methods were successfully implemented following standard algorithms:
   - Explicit Euler: Simple but inadequate for this problem
   - Implicit Euler: Computationally expensive with no accuracy advantage
   - Runge-Kutta 4: Excellent performance with manageable cost
3. **Comparative Analysis**
   Quantitative comparison revealed RK4's superiority:
   - RK4 errors: $O(10^{11})$
   - Euler methods errors: $O(10^{16})$–$O(10^{17})$
   - Improvement factor: $\sim 10^5$ for only 4× computational cost

Theoretical Insights:

- Systems with positive eigenvalues exhibit exponential error amplification that renders first-order methods unsuitable

- A-stability of implicit methods does not prevent error accumulation in unstable systems
- Fourth-order accuracy is essential for problems with significant growth or oscillations
- The local truncation error order directly impacts the ability to track exponentially growing solutions

Practical Implications:

For scientific computing applications involving differential equations with similar characteristics (exponential growth, moderate time scales), the following recommendations apply:

1. Always use at least fourth-order methods (RK4 or higher)
2. Monitor error estimates through comparison with analytical solutions when available
3. Consider adaptive step-size control for long-time integration
4. Reserve implicit methods for truly stiff problems where stability, not just accuracy, is the primary concern

Limitations and Future Work:

This study focused on a single ODE with specific parameters. Future investigations could explore:

- Effect of step size variation on accuracy and computational efficiency
- Performance of other methods (multistep methods, Runge-Kutta-Fehlberg with adaptive steps)
- Extension to systems of coupled ODEs
- Analysis of stiff ODEs where implicit methods excel
- Investigation of symplectic integrators for Hamiltonian systems

Concluding Remarks:

The integration of analytical and numerical approaches provided a complete understanding of both the solution itself and the performance characteristics of different numerical methods. The analytical solution served as an invaluable benchmark, enabling precise quantification of numerical errors. This work demonstrates that method selection must be guided by problem characteristics: for systems with exponential growth, high-order methods are not just preferable but essential for obtaining meaningful results.

The comprehensive documentation in the accompanying `Integrators.ipynb` file and the generated figures provide a complete, reproducible framework for ODE analysis that can be adapted to other problems in numerical analysis and computational science.

BIBLIOGRAPHY

[1] Burden R.L., Faires J.D., Burden A.M. Numerical Analysis. 10th ed. Cengage Learning, 2015.

[2] Press W.H., Teukolsky S.A., Vetterling W.T., Flannery B.P. Numerical Recipes: The Art of Scientific Computing. 3rd ed. Cambridge University Press, 2007.

[3] Butcher J.C. A history of Runge-Kutta methods // Applied Numerical Mathematics. 1996. Vol. 20, No. 3. P. 247–260.

[4] Hairer E., Nørsett S.P., Wanner G. Solving Ordinary Differential Equations I: Nonstiff Problems. 2nd ed. Springer, 1993.